The SIP Debug Output Filtering Support feature provides the capability for SIP-related debug output to be filtered based on a set of user-defined matching conditions.

Guide Contents
Troubleshooting Cisco IOS Voice Overview
Debug Command Output on Cisco IOS Voice Gateways
Filtering Troubleshooting Output
Cisco VoIP Internal Error Codes
Troubleshooting Cisco IOS Voice Telephony
Troubleshooting Cisco IOS Voice Protocols
Troubleshooting Cisco IOS Telephony Applications
Monitoring the Cisco IOS Voice Network
Cause Codes and Debug Values

Contents

Restrictions for SIP Debug Output Filtering Support

The SIP Debug Output Filtering Support feature filters all enabled bugs, not just SIP bugs.

Information About SIP Debug Output Filtering Support

To configure the SIP Debug Output Filtering Support feature, you should understand the following concepts:

- Feature Design of SIP Debug Output Filtering Support

Feature Design of SIP Debug Output Filtering Support

Prior to the SIP Debug Output Filtering Support feature, debugging and troubleshooting on the VoIP gateway was made more challenging by the extensive amounts of raw data generated by debug output. This feature allows the debug output for a SIP call to be filtered according to a variety of criteria. The SIP Debug Output Filtering Support feature provides a generic call filtering mechanism that does the following:

- Allows you to define a set of matching conditions for filtering calls.
- Identifies the desired calls based on the configured matching conditions inside VoIP gateways.
- Coordinates the filtering effort on traced calls between multiple modules inside VoIP gateways.
- Displays the debugging trace for calls that match the specified conditions.

The SIP Debug Output Filtering Support feature document provides SIP-specific information on the debug filtering design in Cisco IOS gateways implemented by the voice call debug filtering feature. See the Voice Call Debug Filtering on Cisco Voice Gateways for more information.

Generic Call Filtering Module

Cisco implements the SIP Debug Output Filtering Support feature using the GCFM to identify and track calls based on configured parameters. You can use the command-line interface (CLI) to turn the GCFM on and off. When the GCFM is turned on, only the debug messages for calls that match the filtering conditions are displayed.

The GCFM manages the set of matching condition lists and coordination and tracking of calls between multiple modules within the voice gateway architecture. The SIP module uses the GCFM function to create GUIDs to track individual inbound and outbound SIP calls. Activity in the GCFM can be traced using the **debug call filter detail** and **debug call filter inout** commands. See the Cisco IOS Debug Command Reference for more information about these debug commands.

SIP Debug Commands that Support Output Filtering

- **debug ccsip all**
- **debug ccsip calls**
- **debug ccsip events**
- **debug ccsip messages**
- **debug ccsip preauth**
- **debug ccsip states**

See the Cisco IOS Debug Command Reference for more information about SIP debug commands.

Note: Because of the importance of the messages associated with the **debug ccsip err** command, this debug output is not filterable.

Matching Conditions

To filter calls, define a list of matching conditions. The SIP Debug Output Filtering Support feature supports matches based on the following conditions:

- Incoming calling-number string

- Incoming called-number string
- Incoming signaling IPv4 local address
- Incoming signaling IPv4 remote address
- Incoming media IPv4 local address
- Incoming media IPv4 remote address
- Incoming dial peer
- Outgoing calling-number string
- Outgoing called-number string
- Outgoing signaling IPv4 local address
- Outgoing signaling IPv4 remote address
- Outgoing media IPv4 local address
- Outgoing media IPv4 remote address
- Outgoing dial peer

The string pattern for calling and called numbers can be either a complete telephone number or a partial telephone number with wildcard digits, represented by a period (.) character.

You can define matching conditions as follows:

- Exact match-All related debug output is filtered until all conditions in the match list are explicitly met. This option is the best choice for most situations because it produces the most concise output.
- Partial match-No related debug output is filtered until there is a single explicit match failure. As long as zero or more conditions are met, debug output is not filtered. This choice is useful in debugging call startup problems like digit collection, but is not ideal for many situations because of the large amount of debug output until matches explicitly fail.

See the Exact and Partial Matching for more information on matching conditions and filtering voice calls.

Configuring SIP Debug Filtering

This section contains the following procedures:

Configuring Call Filters

This task configures the conditions for filtering SIP calls.

SUMMARY STEPS

1. **enable**
2. **configure** terminal
3. **call filter match-list** *number* voice
4. **incoming calling-number** *string*
5. **incoming called-number** *string*
6. **incoming signaling** {**local** | **remote**} **ipv4 ip-address**
7. **incoming media** {**local** | **remote**} **ipv4 ip-address**
8. **incoming dialpeer** *tag*
9. **outgoing calling-number' string**
10. **outgoing called-number** *string*

11. **outgoing signaling** {**local** | **remote**} **ipv4** *ip-address*
12. **outgoing media** {**local** | **remote**} **ipv4** *ip-address*
13. **outgoing dialpeer** *tag*
14. **end**

DETAILED STEPS

	Command or Action	Purpose
1.	**enable** Example: `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
2.	**configure** terminal Example: `Router# configure terminal`	Enters global configuration mode.
3.	**call filter match-list** *number* **voice** Example: `Router(config)# call filter match-list 1 voice`	Enters call filter match list configuration mode to define the filter conditions. • *number*-Numeric label that uniquely identifies the match list. Range is 1 to 16. **Note**: At least one of the following optional parameters (Step 4 to Step 13) for call filtering must be configured.
4.	**incoming calling-number** *string* Example: `Router(conf-call-filter-mlist)# incoming calling-number 408555`	(Optional) Specifies the incoming calling number to be filtered. • *string*-Numeric string that identifies all or part of the incoming calling number.
5.	**incoming called-number** *string* Example: `Router(conf-call-filter-mlist)# incoming called-number 408555`	(Optional) Specifies the incoming called number to be filtered. • *string*-Numeric string that identifies all or part of the incoming called number.
6.	**incoming signaling** {**local** | **remote**} **ipv4** *ip-address* Example:	(Optional) Specifies the incoming signaling IPv4 address.

	`Router(conf-call-filter-mlist)# incoming signaling local ipv4 192.168.10.255`	• **local**-Local voice gateway • **remote**-Remote IP device • *ip-address*-IP address of the local voice gateway.
7.	**incoming media** {**local** | **remote**} **ipv4** *ip-address* Example: `Router(conf-call-filter-mlist)# incoming media local ipv4 192.168.10.255`	(Optional) Specifies the incoming media IPv4 address for the voice gateway receiving the media stream. • **local**-Local voice gateway • **remote**-Remote IP device • *ip-address*-IP address of the local voice gateway.
8.	**incoming dialpeer** *tag* Example: `Router(conf-call-filter-mlist)# incoming dialpeer 14`	(Optional) Specifies the incoming dial peer to be filtered. • *tag*-Digits that define a specific dial peer. Valid entries are 1 to 2147483647.
9.	**outgoing calling-number** *string* Example: `Router(conf-call-filter-mlist)# outgoing calling-number 408555`	(Optional) Specifies the outgoing calling number to be filtered; this number goes out after number translation and expansion are complete. • *string*-Numeric string that identifies all or part of the outgoing calling number.
10.	**outgoing called-number** *string* Example: `Router(conf-call-filter-mlist)# outgoing called-number 408555`	(Optional) Specifies the outgoing called number to be filtered; this number goes out after number translation and expansion are complete. • *string*-Numeric string that identifies all or part of the outgoing called number.
11.	**outgoing signaling** {**local** | **remote**} **ipv4** *ip-address* Example: `Router(conf-call-filter-mlist)# outgoing signaling local`	(Optional) Specifies the outgoing signaling IPv4 address for the gatekeeper managing the signaling.

	`ipv4 192.168.10.255`	• **local**-Local voice gateway • **remote**-Remote IP device • *ip-address*-IP address of the local voice gateway.
12.	**outgoing media** {**local** | **remote**} **ipv4** *ip-address* Example: `Router(conf-call-filter-mlist)# outgoing media local ipv4 192.168.10.255`	(Optional) Specifies the outgoing media IPv4 address for the voice gateway receiving the media stream. • **local**-Local voice gateway • **remote**-Remote IP device • *ip-address*-IP address of the local voice gateway.
13.	**outgoing dialpeer** *tag* Example: `Router(conf-call-filter-mlist)# outgoing dialpeer 14`	(Optional) Specifies the outgoing dial peer to be filtered. • *tag*-Digits that define a specific dial peer. Valid entries range from1 to 2147483647.
14.	**end** Example: `Router(conf-call-filter-mlist)# end`	Exits to privileged EXEC mode.

What to Do Next

After the conditions are set for filtering the SIP call debug output, **debug** commands can be enabled. Proceed to the "Enabling SIP Debug Output Filtering" section.

Enabling SIP Debug Output Filtering

This task enables debug filtering for SIP-related output.

Prerequisites

The conditions for the SIP call debug filter must be set as described in the Configuring Call Filters.

SUMMARY STEPS

1. **enable**
2. **debug condition match-list** *number* {**exact-match** | **partial-match**}
3. **debug ccsip** {**all** | **calls** | **events** | **messages** | **preauth** | **states**}

DETAILED STEPS

	Command or Action	Purpose
1.	**enable** Example: `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
2.	**debug condition match-list number** {**exact-match** | **partial-match**) Example: `Router# debug condition match-list 1 exact-match`	Enables the filter match list for the set conditions. • *number*-Numeric label that uniquely identifies the match list. Range is 1 to 16. • **exact-match**-All related debug output is filtered until all conditions in the match list are explicitly met. This is the best choice for most situations because the output is the most concise. • **partial-match**-No related debug output is filtered until there is a single explicit match failure. As long as zero or more conditions are met, debug output is not filtered. This choice is useful in debugging call startup problems like digit collection, but is not ideal for many situations because a large amount of debug output is generated before matches explicitly fail.
3.	**debug ccsip** {**all** | **calls** | **events** | **messages** | **preauth** | **states**} Example: `Router# debug ccsip all`	Enables the appropriate SIP call debug commands. • See the Cisco IOS Debug Command Reference for detailed descriptions of these debug commands. • The debug output begins at this point.

Verifying SIP Debug Output Filtering Support

Perform this task to verify debug output filtering.

SUMMARY STEPS

1. **show debug**
2. **show call filter components**
3. **show call filter match-list**

DETAILED STEPS

1. **show debug**

Use this command to display the debugs that are enabled, for example:

```
Router# show debug
CCSIP SPI:SIP Call Statistics tracing is enabled       (filter is ON)
CCSIP SPI:SIP Call Message tracing is enabled  (filter is ON)
CCSIP SPI:SIP Call State Machine tracing is enabled    (filter is ON)
CCSIP SPI:SIP Call Events tracing is enabled   (filter is ON)
```

```
CCSIP SPI:SIP error debug tracing is enabled   (filter is ON)
CCSIP SPI:SIP info debug tracing is enabled    (filter is ON)
CCSIP SPI:SIP media debug tracing is enabled   (filter is ON)
CCSIP SPI:SIP Call preauth tracing is enabled (filter is ON)
```

2. **show call filter components**

Use this command to display which components are registered with the GCFM, the internal module that controls which components are filtered:

```
The following components registered in GCFM:
     ISDN
     VTSP
     CCAPI
     TGRM
     DIAL-PEER
     NUMBER-TRANSLATION
     SSAPP
     VOICE-IVR-V2
     H323
     SIP
     CRM
```

3. **show call filter match-list**

Use this command to display the criteria set for the specified match list. The command shows a list of all the match lists, shows which ones are enabled, and shows whether they are enabled for partial or exact matching.

```
Router# show call filter match-list

****************************************
call filter match-list 1 voice
****************************************
  incoming called-number 5551200
```

Configuration Examples for SIP Debug Filtering

This section provides the following configuration examples:

- Configuring Call Filters: Example
- Enabling SIP Debug Output Filtering: Example

Configuring Call Filters: Example

The following example shows how to configure call filters. In the example, the Cisco AS5300 configuration defining a match list and specifying the incoming called number to be filtered is shown.

```
Router# show running-config
Building configuration...
Current configuration :3944 bytes
!
! Last configuration change at 13:07:08 EST Tue Sep 2 2003
!
version 12.3
no service pad
service timestamps debug datetime msec localtime
service timestamps log datetime msec localtime
no service password-encryption
```

```
service internal
!
hostname Router-1
!
logging buffered 500000 debugging
enable password lab
!
!
!
resource-pool disable
clock timezone EST -5
clock summer-time EST recurring
!
no aaa new-model
ip subnet-zero
ip domain name cisco.com
ip host CALLGEN-SECURITY-V2 10.73.75.86 8.52.0.0
ip name-server 10.44.11.21
!
!
isdn switch-type primary-dms100
no scripting tcl init
no scripting tcl encdir
!
!
voice call carrier capacity active
!
voice service voip
 no supplementary-service h450.12
 sip
!
!
voice class codec 1
 codec preference 1 g711ulaw
 codec preference 2 g729r8
!
!
voice hpi capture buffer 10000
no voice hpi capture destination
!
!
fax interface-type modem
call-history-mib retain-timer 500
call-history-mib max-size 500
!
!
controller T1 0
 framing esf
 clock source line primary
 linecode b8zs
 pri-group timeslots 1-24
!
controller T1 1
 framing esf
clock source line secondary 1
 linecode b8zs
 pri-group timeslots 1-24
!
controller T1 2
 framing esf
 linecode b8zs
 pri-group timeslots 1-24
!
controller T1 3
```

```
 framing esf
 linecode b8zs
 pri-group timeslots 1-24
!
!
interface Ethernet0
 no ip route-cache
 no ip mroute-cache
 shutdown
 no cdp enable
!
interface Serial0:23
 no logging event link-status
isdn switch-type primary-dms100
 isdn incoming-voice modem
 no cdp enable
!
interface Serial1:23
 no logging event link-status
 isdn switch-type primary-dms100
 isdn incoming-voice modem
 no cdp enable
!
interface Serial2:23
 no logging event link-status
 isdn switch-type primary-dms100
 isdn incoming-voice modem
 no cdp enable
!
interface Serial3:23
 isdn switch-type primary-dms100
 no cdp enable
!
interface FastEthernet0
 ip address 172.18.195.43 255.255.0.0
 no ip route-cache
no ip mroute-cache
 duplex auto
 speed auto
 fair-queue 64 256 235
 no cdp enable
!
ip classless
ip route 0.0.0.0 0.0.0.0 FastEthernet0
ip route 10.0.0.0 255.0.0.0 172.18.193.3
ip route 10.0.0.0 255.0.0.0 172.18.195.1
ip route 172.18.0.0 255.255.0.0 172.18.193.1
ip http server
!
!
!
control-plane
!
!
!
call filter match-list 1 voice
incoming called-number 4085559876
!
voice-port 0:D
!
voice-port 1:D
!
voice-port 2:D
!
```

```
voice-port 3:D
!
!
dial-peer cor custom
!
!
!
dial-peer voice 1100 pots
 destination-pattern 55511..
 direct-inward-dial
 port 0:D
 prefix 55511
!
dial-peer voice 1200 pots
 destination-pattern 55512..
direct-inward-dial
 port 1:D
 prefix 55512
!
dial-peer voice 444 pots
 destination-pattern 444....
 direct-inward-dial
 port 2:D
 prefix 444
!
dial-peer voice 5100 voip
 destination-pattern 55551..
 session protocol sipv2
 session target ipv4:172.18.207.10:5060
 dtmf-relay rtp-nte
 codec g711ulaw
!
dial-peer voice 5200 voip
 destination-pattern 55552..
 session protocol sipv2
 session target ipv4:172.18.207.10:5060
 dtmf-relay rtp-nte
!
dial-peer voice 5300 voip
 destination-pattern 55553..
 session protocol sipv2
 session target ipv4:10.0.0.5
 dtmf-relay rtp-nte
!
dial-peer voice 5400 voip
 destination-pattern 55554..
 session protocol sipv2
 session target ipv4:10.0.0.5
 dtmf-relay rtp-nte
!
dial-peer voice 2100 voip
 destination-pattern 55521..
 session protocol sipv2
 session target ipv4:172.18.193.88
 dtmf-relay rtp-nte
 codec g711ulaw
!
dial-peer voice 2200 voip
 destination-pattern 55522..
 session protocol sipv2
 session target ipv4:172.18.207.10:5060
dtmf-relay rtp-nte
!
gateway
```

```
!
sip-ua
 retry invite 3
 retry response 3
 retry bye 3
 retry cancel 3
 retry prack 3
 retry comet 3
 retry rel1xx 3
 retry notify 3
 timers trying 750
!
!
line con 0
 exec-timeout 0 0
 transport preferred none
line aux 0
line vty 0 4
 exec-timeout 0 0
 password lab
login
 transport preferred none
!
ntp clock-period 17180086
ntp server 172.68.10.150 prefer
!
end
```

Enabling SIP Debug Output Filtering: Example

The following example shows how to enable SIP debug output filtering. In the following example, the match-list configuration, enabled debugs, and debug output for a Cisco AS5300 are shown.

Note: When the exact match condition is used for SIP call debug filtering, all related debug output is filtered until all conditions in the match list are explicitly met.

```
Router# debug condition match-list 1 exact-match
Router# debug ccsip all
Router# show debug
CCSIP SPI:SIP Call Statistics tracing is enabled       (filter is ON)
CCSIP SPI:SIP Call Message tracing is enabled  (filter is ON)
CCSIP SPI:SIP Call State Machine tracing is enabled    (filter is ON)
CCSIP SPI:SIP Call Events tracing is enabled   (filter is ON)
CCSIP SPI:SIP error debug tracing is enabled   (filter is ON)
CCSIP SPI:SIP info debug tracing is enabled    (filter is ON)
CCSIP SPI:SIP media debug tracing is enabled   (filter is ON)
CCSIP SPI:SIP Call preauth tracing is enabled  (filter is ON)
Router# Debug filtering is now on
Building configuration...
!
!
!
call filter match-list 1 voice
incoming called-number 4085551221
!
end
```

The following lines show partial debug output with the filter on. Some debug output generated during a call may not have GUID information. These debugs, represented by /xxxxxxxxxxx/ characters in the debug header, are not filtered and always appear.

```
Sep  2 13:11:05.395://-1/xxxxxxxxxxx/SIP/Error/httpish_msg_process_network_msg:Content
```

```
Length 222, Bytes Remaining 233
Sep  2 13:11:05.395://-1/xxxxxxxxxxxx/SIP/Info/HandleUdpSocketReads:Msg enqueued for SPI
with IP addr:10.102.16.214:56587
Sep  2 13:11:05.395://-1/xxxxxxxxxxxx/SIP/Info/sipSPISetDateHeader:Converting TimeZone EST
to SIP default timezone = GMT
Sep  2 13:11:05.399://-1/xxxxxxxxxxxx/SIP/Error/sipSPI_validate_own_ip_addr:ReqLine IP
addr does not match with host IP addr
Sep  2 13:11:05.399://-1/xxxxxxxxxxxx/SIP/Event/sipSPIEventInfo:Queued event from SIP SPI
:SIPSPI_EV_SEND_MESSAGE
```

The next lines show the debug filtering turned off.

```
Router# no debug condition match-list 1
Router# show debug
CCSIP SPI:SIP Call Statistics tracing is enabled       (filter is OFF)
CCSIP SPI:SIP Call Message tracing is enabled  (filter is OFF)
CCSIP SPI:SIP Call State Machine tracing is enabled    (filter is OFF)
CCSIP SPI:SIP Call Events tracing is enabled   (filter is OFF)
CCSIP SPI:SIP error debug tracing is enabled   (filter is OFF)
CCSIP SPI:SIP info debug tracing is enabled    (filter is OFF)
CCSIP SPI:SIP media debug tracing is enabled   (filter is OFF)
CCSIP SPI:SIP Call preauth tracing is enabled  (filter is OFF)
```

With filtering turned off, all the debug output is displayed.

```
Sep  2 13:12:31.112://-1/xxxxxxxxxxxx/SIP/Info/HandleUdpSocketReads:Msg enqueued for SPI
with IP addr:10.102.16.214:56589
Sep  2 13:12:31.112://-1/xxxxxxxxxxxx/SIP/Msg/ccsipDisplayMsg:
Received:
SUBSCRIBE sip:3100802@172.18.193.99:5060 SIP/2.0
Via:SIP/2.0/UDP  172.18.193.100:5060;branch=z9hG4bK1348
From:"3100801" <sip:3100801@172.18.193.100>;tag=19AE8-A6C
To:<sip:3100802@172.18.193.99>;tag=19AE8-A6C
Date:Mon, 30 Oct 2000 04:47:31 GMT
Call-ID:96ACB04F-AD5611D4-800894FA-5B905DB@172.18.193.100
Supported:timer,100rel
Min-SE: 1800
Cisco-Guid:2496939846-2908099028-2147680272-2073165500
User-Agent:Cisco-SIPGateway/IOS-12.x
Allow:INVITE, OPTIONS, BYE, CANCEL, ACK, PRACK, COMET, REFER, SUBSCRIBE, NOTIFY, INFO,
UPDATE
CSeq:101 SUBSCRIBE
Max-Forwards:6
Remote-Party-ID:<sip:3100801@172.18.193.100>;party=calling;screen=no;privacy=off
Timestamp:972881251
Contact:<sip:3100801@172.18.193.100:5060>
Expires:60
Allow-Events:telephone-event
Content-Type:application/sdp
Content-Length:233
v=0
o=CiscoSystemsSIP-GW-UserAgent 2904 6070 IN IP4 172.18.193.100
s=SIP Call
c=IN IP4 172.18.193.100
t=0 0
m=audio 18862 RTP/AVP 18 19
c=IN IP4 172.18.193.100
a=rtpmap:18 G729/8000
a=fmtp:18 annexb=no
a=rtpmap:19 CN/8000
a=ptime:20
Sep  2 13:12:31.112://-1/94D447468003/SIP/State/sipSPIChangeState:0x631570C8 :State change
from (STATE_NONE, SUBSTATE_NONE)  to (STATE_IDLE, SUBSTATE_NONE)
```

```
Sep  2 13:12:31.112://-1/xxxxxxxxxxxx/SIP/Info/sipSPISetDateHeader:Converting TimeZone EST
to SIP default timezone = GMT
Sep  2 13:12:31.116://-1/xxxxxxxxxxxx/SIP/Error/sipSPI_validate_own_ip_addr:ReqLine IP
addr does not match with host IP addr
Sep  2 13:12:31.116://-1/xxxxxxxxxxxx/SIP/Event/sipSPIEventInfo:Queued event from SIP SPI
:SIPSPI_EV_SEND_MESSAGE
Sep  2 13:12:31.116://-1/94D447468003/SIP/Info/sipmwiEventCleanup:Removing CCB with mwi
clientID(3100802)
Sep  2 13:12:31.116://-1/94D447468003/SIP/State/sipSPIChangeState:0x631570C8 :State change
from (STATE_IDLE, SUBSTATE_NONE)  to (STATE_DEAD, SUBSTATE_NONE)
Sep  2 13:12:31.116://-1/94D447468003/SIP/Info/ccsip_deregister_call_filter:Deregistering
call from GCFM
Sep  2 13:12:31.120://-1/94D447468003/SIP/Info/sipSPIFlushEventBufferQueue:There are 0
events on the internal queue that are going to be free'd
Sep  2 13:12:31.120://-1/94D447468003/SIP/Info/sipSPIUfreeOneCCB:Freeing ccb 631570C8
Sep  2 13:12:31.120://-1/xxxxxxxxxxxx/SIP/Msg/ccsipDisplayMsg:
Sent:
SIP/2.0 489 Bad Event - 'Missing Event:field'
Via:SIP/2.0/UDP
172.18.193.100:5060;branch=z9hG4bK1348;received=10.102.16.214
From:"3100801" <sip:3100801@172.18.193.100>;tag=19AE8-A6C
To:<sip:3100802@172.18.193.99>;tag=19AE8-A6C
Call-ID:96ACB04F-AD5611D4-800894FA-5B905DB@172.18.193.100
CSeq:101 SUBSCRIBE
Timestamp:972881251
Content-Length:0
```